Yee &
Associates, P.C.

4100 Alpha Road
Suite 1100
Dallas, Texas 75244

Main No. (972) 385-8777
Facsimile (972) 385-7766

Facsimile Cover Sheet

To: Commissioner for Patents for **Examiner Douglas B. Blair** **Group Art Unit 2142**	Facsimile No.: **571/273-8300**
From: Michele Morrow Legal Assistant to Gerald H. Glanzman	No. of Pages Including Cover Sheet: 28

Message:

Enclosed herewith:

* Transmittal of Appeal Brief; and
* Appeal Brief.

Re: Application No. 09/651,585
   Attorney Docket No: JP9-1999-0175US1

Date: Monday, May 08, 2006

Please contact us at (972) 385-8777 if you do not receive all pages indicated above or experience any difficulty in receiving this facsimile.	*This Facsimile is intended only for the use of the addressee and, if the addressee is a client or their agent, contains privileged and confidential information. If you are not the intended recipient of this facsimile, you have received this facsimile inadvertently and in error. Any review, dissemination, distribution, or copying is strictly prohibited. If you received this facsimile in error, please notify us by telephone and return the facsimile to us immediately.*

PLEASE CONFIRM RECEIPT OF THIS TRANSMISSION BY FAXING A CONFIRMATION TO 972-385-7766.

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re application of: **Inagaki et al.**	§	Group Art Unit: **2142**
	§	
Serial No.: **09/651,585**	§	Examiner: **Blair, Douglas B.**
	§	
Filed: **August 29, 2000**	§	Attorney Docket No.: **JP9-1999-0175US1**
	§	
For: **Client Server System and Method**	§	
for Executing an Application Utilizing	§	
Distributed Objects	§	

36736

PATENT TRADEMARK OFFICE
CUSTOMER NUMBER

> **Certificate of Transmission Under 37 C.F.R. § 1.8(a)**
> I hereby certify this correspondence is being transmitted via facsimile to the Commissioner for Patents, P.O. Box 1450, Alexandria, VA 22313-1450, facsimile number (571) 273-8300 on May 8, 2006.
>
> By: *Michele Morrow*
> Michele Morrow

TRANSMITTAL OF APPEAL BRIEF

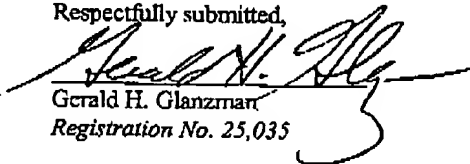Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

Sir:
ENCLOSED HEREWITH:

* Appeal Brief (37 C.F.R. 41.37)

A fee of $500.00 is required for filing an Appeal Brief. Please charge this fee to IBM Corporation Deposit Account No. 09-0461. No additional fees are believed to be necessary. If, however, any additional fees are required, I authorize the Commissioner to charge these fees which may be required to IBM Corporation Deposit Account No. 09-0461. No extension of time is believed to be necessary. If, however, an extension of time is required, the extension is requested, and I authorize the Commissioner to charge any fees for this extension to IBM Corporation Deposit Account No. 09-0461.

Respectfully submitted,

Gerald H. Glanzman
Registration No. 25,035

Duke W. Yee
Registration No. 34,285
YEE & ASSOCIATES, P.C.
P.O. Box 802333
Dallas, Texas 75380
(972) 385-8777
ATTORNEYS FOR APPLICANTS

Docket No. JP9-1999-0175US1 *PATENT*

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re application of: **Inagaki et al.** §	
§	Group Art Unit: **2142**
Serial No. **09/651,585** §	
§	Examiner: **Blair, Douglas B.**
Filed: **August 29, 2000** §	
§	
For: **Client Server System and Method** §	
for Executing an Application Utilizing §	
Distributed Objects §	

Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

36736
PATENT TRADEMARK OFFICE
CUSTOMER NUMBER

APPEAL BRIEF (37 C.F.R. 41.37)

This brief is in furtherance of the Notice of Appeal, filed in this case on March 6, 2006.

A fee of $500.00 is required for filing an Appeal Brief. Please charge this fee to IBM Corporation Deposit Account No. 09-0461. No additional fees are believed to be necessary. If, however, any additional fees are required, I authorize the Commissioner to charge these fees which may be required to IBM Corporation Deposit Account No. 09-0461. No extension of time is believed to be necessary. If, however, an extension of time is required, the extension is requested, and I authorize the Commissioner to charge any fees for this extension to IBM Corporation Deposit Account No. 09-0461.

05/09/2006 HGUTEMA1 00000015 090461 09651585
01 FC:1402 500.00 DA

REAL PARTY IN INTEREST

The real party in interest in this appeal is the following party: International Business Machines Corporation

RELATED APPEALS AND INTERFERENCES

With respect to other appeals or interferences that will directly affect, or be directly affected by, or have a bearing on the Board's decision in the pending appeal, there are no such appeals or interferences.

STATUS OF CLAIMS

A. TOTAL NUMBER OF CLAIMS IN APPLICATION

Claims in the application are: 1-15

B. STATUS OF ALL THE CLAIMS IN APPLICATION

1. Claims canceled: NONE
2. Claims withdrawn from consideration but not canceled: NONE
3. Claims pending: 1-15
4. Claims allowed: NONE
5. Claims rejected: 1-15
6. Claims objected to: NONE

C. CLAIMS ON APPEAL

The claims on appeal are: 1-15

STATUS OF AMENDMENTS

A Response to the Final Office Action dated December 29, 2005, was not filed. Accordingly, the claims on appeal herein are as amended in the Response to Office Action dated October 13, 2005.

(Appeal Brief Page 5 of 26)
Inagaki et al. – 09/651,585

SUMMARY OF CLAIMED SUBJECT MATTER

A.   CLAIM 1 - INDEPENDENT

The subject matter of claim 1 is directed to client server system that uses distributed objects (see page 12, lines 3-11 in conjunction with **Figure 1** and page 27, lines 10-18 in conjunction with **Figure 21**). A client **73** is connected to a communication network **74, 75** (see page 27, lines 13-18 and **Figure 21**) for performing an access request to an object. An application server **20 or 71** (see page 12, lines 6-7 in conjunction with **Figure 1** and page 27, lines 11-13 in conjunction with **Figure 21**) performs an application by an actual object **16** (see page 12, line 10 and **Figure 1**) according to the access request by the client (reference number **11** in **Figure 1** and reference number **73** in **Figure 21**). An object pool server **72** (see page 27, lines 13-15 and **Figure 21**) is connected to the client **73** through the communication network **74, 75** and is connected to the application server **71** for pooling a proxy object **17** (see page 12, lines 10-11 and **Figure 1**) corresponding to the actual object **16** and for holding actual object management information that is information on the actual object **16**. The application server **71** notifies the object pool server **72** of an event according to a change in status of the application and the object pool server **72** automatically updates the actual object management information according to the notification of the event from the application server **71** (see, for example, page 21, lines 1-7 and **Figure 13**, steps **131** and **132**).

B.   CLAIM 3 – INDEPENDENT

The subject matter of claim 3 is directed to an object pool that uses distributed objects. The object pool **30** includes a client request analyzing unit **41** (see page 15, lines 1-3 and **Figure 5** for analyzing an access request to an object, and an object information storage unit **35** (see page 14, lines 14-16 and **Figure 5**) for storing object information at a termination process of the object pool. An object creating unit **42** (see page 15, lines 2-3 and **Figure 5**) creates an object at a starting process of the object pool according to the object information stored by the object information storage unit **35**, and an object managing unit **32** (see page 14, lines 9-12 and **Figure 5**) pools the object created by the object creating unit **42** before accessing the object from the client.

C. CLAIM 5 – INDEPENDENT

The subject matter of claim 5 is directed to a client server system using distributed objects. The system includes an object pool **30** (see page 12, lines 8-10 in conjunction with **Figure 1**, and page 14, line 9 to page 15, line 15 in conjunction with **Figure 5**) connected through a communication network **74, 75** (see page 27, lines 13-18 and **Figure 21**) to a client **11, 73** (see page 12, lines 4-6 in conjunction with **Figure 1**, and page 27, line 15 in conjunction with **Figure 21**) which sends an access request to an object, and for pooling objects and managing object information. An application execution environment **20** (see page 12, lines 6 and 7 and **Figure 1**) is connected to the object pool **30** for executing an application according to the access request of the client **11**, and notifies the object pool **30** of an event according to a status change of the application, wherein the object pool **30** updates the object information according to the event notification from the application execution environment **20**.

D. CLAIM 8 – INDEPENDENT

The subject matter of claim 8 is directed to an object pooling method for pooling objects in advance on a specified server to execute an application in a distributed system. The object pooling method comprises the steps of storing object information when a process of the server is terminated, creating objects according to the object information when a process of the server is started, and pooling the created objects (see page 21, line 21 to page 23, line 5 and **Figures 14 and 15**).

E. CLAIM 10 – INDEPENDENT

The subject matter of claim 10 is directed to an object pooling method on an object pool which is connected through a communication network to a client which sends an access request to an object, and is connected to an application execution environment in which an actual object is executed. The method comprises the steps of pooling a proxy object corresponding to an actual object in the application execution environment, recognizing a change in status of the actual object in the application execution environment, and updating actual object management

information according to the recognized status change (see page 20, line 17 to page 21, line 20 and **Figure 13**).

F.   CLAIM 11 – INDEPENDENT

The subject matter of claim 11 is directed to a storage medium in which a program to be executed by a computer is stored so that it can be read by an input unit of the computer. The program makes the computer execute a monitoring process for monitoring execution status of a project which executes an application utilizing distributed objects, an event creating process for creating an event according to execution status of the project monitored by the monitoring process (see page 13, lines 4-5 and **Figure 2**), and an event issuing process for issuing the event created by the event creating process to an object pool server connected through a network (see page 13, lines 5-7 and **Figure 2**).

G.   CLAIM 12 – INDEPENDENT

The subject matter of claim 12 is directed to a storage medium in which a program to be executed by a computer is stored so that it can be read by an input unit of the computer. The computer is connected through a network 74, 75 (see page 27, lines 13-18 and **Figure 21**) or the like to an application server 71 for executing an application utilizing distributed objects, and to a client which requests the execution of the application. The program makes the computer execute an object pooling process for pooling proxy objects corresponding to actual objects 16 to be executed by the application server (see page 12, lines 10-11 and **Figure 1**), and an updating process for receiving an event issued from the application server 71 and updating management information on the actual objects 16 according to the event (see page 13, lines 15-24 and **Figure 3**, and page 21, lines 1-7 and **Figure 13**, steps 131 and 132).

H.   CLAIM 14 – INDEPENDENT

The subject matter of claim 14 is directed to a program sending apparatus. The apparatus comprises a storage unit 35 (see page 14, lines 14-16 and **Figure 5**) for storing a software product which makes a computer execute an event forming program for forming an event according to a change in status of an application utilizing distributed objects, and an object

pooling program for pooling objects according to the event formed by the event forming process, and a sending unit for reading out the program from the storage unit, and sending the software product (see, generally, page 30, line 14 to page 31, line 8).

I.   CLAIM 15 – INDEPENDENT

The subject matter of claim 15 is directed to a program sending apparatus. The apparatus comprises a storage unit 35 (see page 14, line 14-16 and **Figure 5**) for storing a program which makes a computer execute an object pooling process for pooling, on a server, objects associated with execution of an application utilizing distributed objects, an information storing process for storing object information in the server, and a creation sequence determining process for determining a sequence of objects to be created according to the object information stored by the information storing process. A sending unit reads out the program from the storage unit, and sends the program (see, generally, page 30, line 14 to page 31, line 8).

J.   CLAIM 9 – DEPENDENT

The subject matter of claim 9, which depends from claim 8, is directed to the object information being stored with a predetermined priority, and objects being created in descending order with respect to the priority (see page 16, lines 1-16 and **Figure 7**).

GROUNDS OF REJECTION TO BE REVIEWED ON APPEAL

A. GROUND OF REJECTION 1 (Claims 1-15)

Claims 1-15 stand rejected under 35 U.S.C. § 102(e) as anticipated by Eytchison (U.S. Patent No. 6,363,434.

ARGUMENT

A.   GROUND OF REJECTION 1 (Claims 1-15)

The Examiner has rejected claims 1-15 under 35 U.S.C. § 102(e) as being anticipated by Eytchison (U.S. Patent No. 6,363,434). This rejection is respectfully traversed.

A.1.   Claims 1, 2, 5-7, 10, 12, 13 and 14

In rejecting the claims, the Examiner states:

> As to claim 1, Eytchison teaches a client server system using distributed objects, comprising: a client connected to a communication network for performing an access request to an object (col. 6, line 64-col. 7, line 11); an application server for performing an application by an actual object according to the access request by said client (col. 6, line 64-col. 7, line 11, the proxies on the home server the application); and an object pool server connected to said client through said communication network and connected to said application server for pooling a proxy object corresponding to said actual object and for holding actual object management information that is part of said actual object (col. 7, lines 20-29, the resource manager pools representations of the proxy devices), wherein said application server notifies said object pool server of an event according to a change in status of said application, and said object pool server automatically updates said actual object management information according to the notification of said event from said application server (col. 7, lines 20-28).

Final Office Action dated December 29, 2005, pages 2-3.

Claim 1 of the present application is as follows:

> 1.   A client server system using distributed objects, comprising:
> a client connected to a communication network for performing an access request to an object;
> an application server for performing an application by an actual object according to the access request by said client; and
> an object pool server connected to said client through said communication network and connected to said application server for pooling a proxy object corresponding to said actual object and for holding actual object management

information that is information on said actual object, wherein said application server notifies said object pool server of an event according to a change in status of said application, and said object pool server automatically updates said actual object management information according to the notification of said event from said application server.

A prior art reference anticipates a claimed invention under 35 U.S.C. § 102 only if every element of the claimed invention is identically shown in that single prior art reference, arranged as they are in the claims. *In re Bond*, 910 F.2d 831, 832, 15 U.S.P.Q.2d 1566, 1567 (Fed. Cir. 1990). All limitations of a claimed invention must be considered when determining patentability. *In re Lowry*, 32 F.3d 1579, 1582, 32 U.S.P.Q.2d 1031, 1034 (Fed. Cir. 1994). Anticipation focuses on whether a claim reads on the product or process a prior art reference discloses, not on what the reference broadly teaches. *Kalman v. Kimberly-Clark Corp.*, 713 F.2d 760, 218 U.S.P.Q. 781 (Fed. Cir. 1983).

Appellants respectfully submit that Eytchison does not identically show every element of the claimed invention arranged as they are in the claims; and, accordingly, does not anticipate the claims. With respect to claim 1, in particular, Eytchison does not teach or suggest "A client server system using distributed objects", and does not teach or suggest such a system that includes "an application server for performing an application by an actual object according to the access request by said client", or "an object pool server connected to said client through said communication network and connected to said application server for pooling a proxy object corresponding to said actual object and for holding actual object management information that is information on said actual object, wherein said application server notifies said object pool server of an event according to a change in status of said application, and said object pool server automatically updates said actual object management information according to the notification of said event from said application server".

Eytchison describes a system for managing consumer electronic media devices such as a DVD player, a TV, a VCR and the like. The system includes a plurality of software device proxies 370a-370i in a home server 214 that control the various electronic devices. A resource manager 320 in the home server determines the availability of particular devices, available bandwidth, and the like, and transmits the necessary commands to the device proxies to control the electronic devices.

(Appeal Brief Page 12 of 26)
Inagaki et al. – 09/651,585

PAGE 14/28 * RCVD AT 5/8/2006 3:25:34 PM [Eastern Daylight Time] * SVR:USPTO-EFXRF-2/17 * DNIS:2738300 * CSID:9723857766 * DURATION (mm-ss):07-00

Eytchison does not disclose "A client server system using distributed objects" as recited in claim 1. As described for example, at page 1, lines 14-18 of the present specification:

> Distributed objects are made from data, applications, or processes and the like, and can be accessed from any place in a network and used for tasks. More specifically, they enable an object oriented method call to be remotely operated, and constructed as middleware implementing an object-oriented programming interface in the upper layer of the socket level.

The electronic devices in Eytchison are physical devices and are not distributed objects used in a client server system as recited in claim 1, as defined in the present specification and as generally recognized by those skilled in the art.

For substantially the same reason, Eytchison does not disclose proxy objects as recited in claim 1. Specifically, as explained at page 1, line 18 to page 2, line 2 of the specification:

> In the distributed object programming technique, an object in a remote machine appears in the local machine as if it were a mirage, and the remote object can be freely manipulated by operating on it. The distributed object mirage is called a proxy object (agent object), which is apparently the same as the object on the remote machine (actual object: remote object), but it exists as an agent rather than a real object, and the first access point for the client is a Web server instead of the actual server. (Emphasis added.)

The device proxies 370a-370i in Eytchison are not the same as the proxy object recited in claim 1. In Eytchison, the device proxies are software in home server 214 that control the physical electronic devices. They are not "mirages" of the electronic devices. This fact is clearly described in col. 7, lines 4-11 of Eytchison, referred to by the Examiner:

> If the devices and the bandwidth are available, the resource manager 320 will return a "granted" signal and transmits the necessary control commands to the software device proxies 370a-370i. The software device proxies 370a-370i then control the devices via IEEE 1394 bus interface 380. (Emphasis added.)

In responding to Appellants arguments made in the Response to Office Action dated October 13, 2005, the Examiner states:

> As to point a), the applicant's specification provides no details as to any real world applications to which the disclosure is directed leaving the terms "object" and "pooling" to be interpreted broadly. Eytchinson's invention is embodied on a home

server so clearly it is a client server system and the devices can be considered distributed objects because they are not part of the server.

Final Office Action dated December 29, 2005, page 5.

Appellants respectfully disagree. As indicated above, the present specification clearly defines what is meant by the terms "distributed object" and "proxy object". Eytchison does not disclose or suggest a system that includes a distributed object and a proxy object as defined in the specification and as recognized by those skilled in the art, and does not anticipate claim 1.

Eytchison also does not disclose or suggest "an application server for performing an application by an actual object according to the access request by said client", or "an object pool server connected to said client through said communication network and connected to said application server for pooling a proxy object corresponding to said actual object and for holding actual object management information that is information on said actual object, wherein said application server notifies said object pool server of an event according to a change in status of said application, and said object pool server automatically updates said actual object management information according to the notification of said event from said application server" as recited in claim 1.

Eytchison discloses a client server 214 that includes all the components illustrated in Figure 3 of Eytchison. On page 5 of the Final Office Action, the Examiner indicates that user application 310 in Eytchison is construed as being the application server in claim 1, and that resource manager 320 in Eytchison is considered to be the object pool server in claim 1. Appellants respectfully disagree.

As described in Col. 6, line 50-Col. 7, line 11 of Eytchison:

> Significantly, according to the present invention, user application 310 is not allowed to communicate directly with software device proxies 370. Rather, user application 310 communicates to the resource manager 320 when it intends to access one of the network consumer electronic media devices. Particularly, user application 310 receives instructions from a user or other entities for a media service, and converts the instructions into a request that identifies the necessary resources for providing the media service. In the following discusion, a media service is defined as content that is displayed or actions that are performed on behalf of the users. For example, an external sensor triggering digital video camera 210 to capture video would be considered a media

<div align="right">
(Appeal Brief Page 14 of 26)
Inagaki et al. – 09/651,585
</div>

service.

In the present embodiment, the user application 310 of FIG. 3 sends the request to the resource manager 320 in the form of an event list that indicates the source device, the destination device and the requested action. Thereafter, the resource manager 320 determines the availability of the source and destination devices, and checks whether sufficient bandwidth is available for carrying out the requested action. If the devices and the bandwidth are available, the resource manager 320 will return a "granted" signal and transmits the necessary control commands to the software device proxies 370a-370i. The software device proxies 370a-370i then control the devices via IEEE 1394 bus interface 380. If the devices or the requisite bandwidth are not available, the resource manager 320 will return a "denied" signal to the user application 310.

As described above, user application 310 in Eytchison receives instructions from a user or other entities for a media service, and converts the instructions into a request that identifies the necessary resources for providing the media service. The user application sends the request to the resource manager 320 in the form of an event list that indicates the source device, the destination device and the requested action, and resource manager 320 determines the availability of the source and destination devices. User application 310 and resource manager 320 in Eytchison are not the same as, and do not perform the same function as, "an application server for performing an application by an actual object according to the access request by said client", and "an object pool server connected to said client through said communication network and connected to said application server for pooling a proxy object corresponding to said actual object and for holding actual object management information that is information on said actual object, wherein said application server notifies said object pool server of an event according to a change in status of said application, and said object pool server automatically updates said actual object management information according to the notification of said event from said application server" as are recited in claim 1.

For at least all the above reasons, claim 1 is not anticipated by Eytchison and patentably distinguishes over Eytchison in its present form..

Claim 2 depends from and further restricts claim 1 and is also not anticipated by Eytchison, at least by virtue of its dependency.

Independent claim 5 is directed to a client server system using distributed objects, and is not anticipated by Eytchison for similar reasons as discussed above with respect to claim 1.

Claims 6 and 7 depend from and further restrict claim 5 and are also not anticipated by

Eytchison, at least by virtue of their dependency.

Independent claims 10, 12 and 14 are not anticipated by Eytchison for similar reasons as discussed above with respect to claim 1. Claim 13 depends from claim 12 and is not anticipated by Eytchison, at least by virtue of its dependency.

A.2. Claims 3, 4, 8, 11 and 15

Independent claim 3 is directed to an object pool that uses distributed objects and is as follows:

> 3. An object pool using distributed objects, comprising:
> a client request analyzing unit for analyzing an access request to an object;
> an object information storage unit for storing object information at a termination process of said object pool;
> an object creating unit for creating an object at a starting process of said object pool according to said object information stored by said object information storage unit; and
> an object managing unit for pooling the object created by said object creating unit before accessing said object from said client.

Eytchison does not disclose an object pool using distributed objects as discussed above with respect to claim 1. In addition, Eytchison does not disclose "an object creating unit for creating an object at a starting process of said object pool according to said object information stored by said object information storage unit", and "an object managing unit for pooling the object created by said object creating unit before accessing said object from said client". The Examiner refers to col. 7, lines 12-28 of Eytchison as disclosing these features. Appellants respectfully disagree. Col. 7, lines 12-28 of Eytchison reads as follows:

> Importantly, according to the present invention, resource manager 320 allows the resources of home network 200 to be checked-in or checked-out independent of application requests. At any time, the resource manager 320 can reclaim checked-out resources and reallocate them to other users. Likewise, a reserved resource can be reclaimed and reallocated to other users or reallocate them back into the resource pool 340.
> According to the present invention, the resource manager 320 of FIG. 3, upon initialization of the home server 214, scans the home network 200 and determines all the

available resources. Data representative of the routing paths and their bandwidths are then stored within path database **330**. Data representative of the available devices are stored within resource pool **340**. As the resources of the home network **200** changes, the resource manager **320** modifies the path database **330** and the resource pool accordingly.

This recitation in Eytchison nowhere discloses "an object creating unit for creating an object at a starting process of said object pool according to said object information stored by said object information storage unit", nor "an object managing unit for pooling the object created by said object creating unit before accessing said object from said client". The recitation talks only about checking in or checking out electronic devices, not about creating an object or pooling the created object as recited in claim 3.

For at least all the above reasons, claim 3 is also not anticipated by Eytchison and patentably distinguishes thereover in its present form.

Claim 4 depends from and further restricts claim 3 and is also not anticipated by Eytchison, at least by virtue of its dependency.

Independent claim 8 is directed to an object pooling method, and is also not anticipated by Eytchison for similar reasons as discussed above with respect to claim 3.

Independent claims 11 and 15 are also not anticipated by Eytchison for similar reasons as discussed above with respect to claim 3.

A.3. Claim 9

Claim 9 depends from claim 8 and is as follows:

> 9. The object pooling method of Claim 8, wherein said object information is stored with a predetermined priority, and said objects are created in descending order with respect to said priority.

The Examiner refers to Col. 9, lines 40-56 of Eytchison as disclosing this feature. Col. 9, lines 40-56 of Eytchison reads as follows:

> Another responsibility of the resource manager **320** is to perform conflict resolutions. If a user with a higher privilege wants to access a service originating from a single threaded device that is in use by another user with a lower privilege, the resource

manager 320 attempts to resolve the conflict. It will send out a message informing the (source/destination) device is in use, and queries the more privileged user whether he/she desires to override the on-going service. A message notifying the user with the lower privilege may be sent indicating that their service is being terminated. When resources become available, the user with the lower privilege is free to re-reschedule the service. As long as there are limited resources and multiple service requests, only the service request with a higher priority will be serviced. In cases where multiple services with identical priorities request the same single threaded resources, a first come first serve policy will be observed.

The above recitation relates to privileges of users. Eytchison describes how a user having a higher privilege will be given access to a device before a user having a lower privilege. The reference nowhere discusses object information being stored with a predetermined priority and objects being "created in descending order with respect to said priority" as recited in claim 9. Objects are not even "created" in Eytchison.

Claim 9, accordingly, is not anticipated by Eytchison in its own right as well as by virtue of its dependency.

For all the above reasons, claims 1-15 are not anticipated by Eytchison, and it is respectfully requested that the Board so find and reverse the Examiner's Final Rejection of the claims.

Gerald H. Glanzman
Reg. No. 25,035
YEE & ASSOCIATES, P.C.
PO Box 802333
Dallas, TX 75380
(972) 385-8777

CLAIMS APPENDIX

The text of the claims involved in the appeal are:

1. A client server system using distributed objects, comprising:

a client connected to a communication network for performing an access request to an object;

an application server for performing an application by an actual object according to the access request by said client; and

an object pool server connected to said client through said communication network and connected to said application server for pooling a proxy object corresponding to said actual object and for holding actual object management information that is information on said actual object, wherein said application server notifies said object pool server of an event according to a change in status of said application, and said object pool server automatically updates said actual object management information according to the notification of said event from said application server.

2. The client server system as set forth in Claim 1, wherein the event notified from said application server is formed according to at least one of a result of a process of starting a project and a result of a process of stopping the project.

3. An object pool using distributed objects, comprising:

a client request analyzing unit for analyzing an access request to an object;

(Appeal Brief Page 19 of 26)
Inagaki et al. – 09/651,585

PAGE 21/28 * RCVD AT 5/8/2006 3:25:34 PM [Eastern Daylight Time] * SVR:USPTO-EFXRF-2/17 * DNIS:2738300 * CSID:9723857766 * DURATION (mm-ss):07-00

an object information storage unit for storing object information at a termination process

of said object pool;

an object creating unit for creating an object at a starting process of said object pool

according to said object information stored by said object information storage unit; and

an object managing unit for pooling the object created by said object creating unit before

accessing said object from said client.

4. The object pool as set forth in Claim 3, wherein the object information stored by said

object information storage unit is constructed so that it can be at least recognized to be a last

accessed object, and said object creating unit starts creation from said last accessed object.

5. A client server system using distributed objects, comprising:

an object pool connected through a communication network to a client which sends an

access request to an object, and for pooling objects and managing object information; and

an application execution environment connected to said object pool for executing an

application according to the access request of said client, and notifying said object pool of an

event according to a status change of said application, wherein said object pool updates said

object information according to the event notification from said application execution

environment.

6. The client server system as set forth in Claim 5, wherein an object pool server having a

function of said object pool and an application server in said application execution environment

are connected to each other through a network, said object pool server pooling objects as proxy objects.

7. The client server system as set forth in Claim 5, wherein said object pool and said application execution environment are formed on the same server.

8. An object pooling method for pooling objects in advance on a specified server to execute an application in a distributed system, said object pooling method comprising the steps of:

storing object information when a process of said server is terminated; and

creating objects according to said object information when a process of said server is started; and

pooling the created objects.

9. The object pooling method of Claim 8, wherein said object information is stored with a predetermined priority, and said objects are created in descending order with respect to said priority.

10. An object pooling method on an object pool which is connected through a communication network to a client which sends an access request to an object, and is connected to an application execution environment in which an actual object is executed, comprising the steps of:

pooling a proxy object corresponding to an actual object in said application execution environment;

recognizing a change in status of the actual object in said application execution environment; and

updating actual object management information according to said recognized status change.

11. A storage medium in which a program to be executed by a computer is stored so that it can be read by an input unit of said computer, wherein said program makes said computer execute:

a monitoring process for monitoring execution status of a project which executes an application utilizing distributed objects;

an event creating process for creating an event according to execution status of the project monitored by said monitoring process; and

an event issuing process for issuing the event created by said event creating process to an object pool server connected through a network.

12. A storage medium in which a program to be executed by a computer is stored so that it can be read by an input unit of said computer, said computer being connected through a network or the like to an application server for executing an application utilizing distributed objects, and to a client which requests the execution of said application, wherein said program makes said computer execute:

an object pooling process for pooling proxy objects corresponding to actual objects to be executed by said application server; and

an updating process for receiving an event issued from said application server, and updating management information on said actual objects according to said event.

13. The storage medium as set forth in Claim 12, wherein said computer is made to further execute:

an execution status managing process for keeping track of execution statuses of said objects, and managing them for each object;

a request analyzing process for analyzing a request including an object creation request and/or an object deletion request from said client; and

a process for executing, upon receipt of a result of a request analysis by said request analyzing process, creation of an object and/or the deletion of an object according to a management result by said execution status managing process.

14. A program sending apparatus, comprising:

a storage unit for storing a software product which makes a computer execute an event forming program for forming an event according to a change in status of an application utilizing distributed objects, and an object pooling program for pooling objects according to the event formed by said event forming process; and

a sending unit for reading out said program from said storage unit, and sending said software product.

15.   A program sending apparatus, comprising:

a storage unit for storing a program which makes a computer execute an object pooling

process for pooling, on a server, objects associated with execution of an application utilizing

distributed objects, an information storing process for storing object information in said server,

and a creation sequence determining process for determining a sequence of objects to be created

according to said object information stored by said information storing process; and

a sending unit for reading out said program from said storage unit, and sending said

program.

EVIDENCE APPENDIX

There is no evidence to be presented.

(Appeal Brief Page 25 of 26)
Inagaki et al. – 09/651,585

RELATED PROCEEDINGS APPENDIX

There are no related proceedings.